What Is The Well-Dressed AI Educator Wearing Now?

WHAT IS THE WELL-DRESSED AI EDUCATOR WEARING NOW?

Alan Bundy
Department of Artificial Intelligence
University of Edinburgh
Edinburgh, UK

A funny thing happened to me at IJCAI-81. I went to a panel on "Education in AI" and stepped back into an argument that I had thought settled several years ago The debate was between the 'scruffies', led by Roger Schank and Ed Feigenbaum, and the 'neats', led by Nils Nilsson.*[1] The neats argued that no education in AI was complete without a strong theoretical component, containing, for instance, courses on predicate logic and automata theory. The scruffies maintained that such a theoretical component was not only unnecessary, but harmful.

The debate stems, of course, from a disagreement on research methodology. The end product of the scruffy researcher is a working computer program, whereas the neat researcher is not satisfied until he has abstracted a theory from the program. Without such a theory, say the neats, AI cannot progress, because achievements cannot be communicated from lab to lab. The programs themselves can rarely be inherited, and then only within a tight knit group. But the scruffies object that there is no *a priori* reason to suppose that such neat theories can be formed. In fact, human intelligence seems not to be like that -- the neats are like the drunk searching under the lamppost. If AI knowledge can be inherited only via working programs, AI can be taught only be apprenticeship, passed from the master to a small band of novices

Both attitudes are harmful. If AI is taught by apprenticeship, it will be divided into small schools with little communication among them, except at the level of slogans. Furthermore, it can be taught only on a small scale to highly skilled programmers. On the other hand, a theoretical approach will exclude large areas of the field as being 'beyond the pale', and is restricted to students with a strong mathematical background. Under either scheme, AI can be taught only to graduates or to senior undergraduates, as part of a final year option. Fortunately, neither attitude is correct, although both contain elements of the truth.

In 1974, a Department of Artificial Intelligence was formed at Edinburgh University and charged with the task of teaching AI courses to undergraduates For non-academic reasons these students were juniors (18/19 years old) and drawn from all parts of the University -- majoring in Linguistics, Psychology, or Philosophy, as well as in Computer Science, Physics, or Mathematics. Simultaneously, similar developments were taking place at Sussex University and the Open University. This was obviously a great challenge and, for the reasons outlined above, some said it was impossible Seven years later, the critics are confounded: the courses are an undoubted success. How is this possible?

The key to the success of the courses is the development of an alternative AI methodology, which we will characterize as 'smart, but casual'. This methodology involves separating out from the profusion of AI programs a central core of techniques and processes. It involves reconstructing the classic works of AI, throwing away the incidental and *ad hoc*, and reworking the remainder in the light of later insight. Such 'rational reconstructions' are presented, without regard to historical niceties, in a way that highlights their essential contribution to AI Doing this is hard (you have to be smart!), and one cannot expect to do it perfectly the first (or even the tenth) time.

This methodology is in sharp contrast to the usual, research oriented, methodologies. Researchers are usually anxious to emphasize the differences between their own work and the work of others. Postgraduate courses often present the latest research without placing it

[1]This terminology was recently introduced, by Roger Schank, to describe the major methodological camps in AI.

in the context of previous work. Junior undergraduate teaching should try to present a coherent subject, and it can do this only by finding the common threads in apparently disparate research and revealing the common basis on which current research is built. It is possible to do this in an elementary AI course, because AI researchers indeed build their work on a common basis, but this basis is usually considered to be too elementary or 'obvious' to be emphasized in a more advanced course.

AI is now seen to have a theoretical component, but the 'theories' are not rigorous mathematical theories, rather they are more casual, code-free descriptions of computational techniques and procedures. The most well known examples are research techniques, like mini-maxing and heuristic search, or parsing algorithms. Such techniques are already contained in books by well known neats, where they are described with the aid of nodes, arcs, numbers, etc., and English text. But other examples can be, and must be, drawn from the scruffier reaches of AI, for instance, script invocation and concept learning.

However, no AI education is complete without acquiring the skill of turning these code-free descriptions into running computer programs. Students must be shown programs which implement the techniques, and they must be able to experiment with them -- adding to them, modifying them, and building alternative versions The traditional AI programming languages -- LISP, POP2, and LOGO -- are unsuitable for this purpose; the gap between the primitives provided by the languages and the techniques to be modelled is too wide. This causes the programs to be too large, since they have to contain lots of basic machinery to support the technique being illustrated. Not only do students have more code to understand, but they have more programming techniques to learn before they can come to grips with AI programs. Senior undergraduate and postgraduates in Computer Science are equipped to handle this burden, but junior undergraduates in Linguistics are not.

The teaching debate in Britain has focussed on providing a programming language which bridges the gap between the primitives and the techniques to be modelled. Most solutions have centered around taking a traditional language and adding packages to it to form new primitives. The best example of this is the Sussex POP11 system, which builds an inference system, parser, linefinder, etc., onto a basic POP2 framework, together with extensive online documentation, to form a self-contained teaching system. The Open University has combined the semantic net language, SOL, with LOGO to form SOLO. At Edinburgh we have become dissatisfied with our attempts to bolt an inference package onto LOGO and have started to experiment with PROLOG. Using these languages, computationally inexperienced students have been able to build non-trivial AI programs within weeks of being introduced to the computer. The debate about which of these languages is best is much healthier than the scruffy/neat debate described above: (a) because it is possible to measure the success of a language, and (b) because the outcome is a range of high-level languages suitable for use by computationally inexperienced students/schoolchildren, etc.

The smart, but casual methodology has a number of benefits:

- □ It produces 'theories' in areas of AI previously dismissed as irredeemably scruffy. Because these theories are casual, they are accessible to the non-mathematically inclined students.
- □ It produces programs which are clear and small, and hence intelligible to students, and even to researchers in other labs, who may then build on them.
- □ It finds common threads in the work of different researchers and places new work in the context of previous work, giving more unity to the field
- □ It makes AI accessible to people who do not have a strong computer science background.

So teaching AI at the junior undergraduate level is not only possible but beneficial to the field, because it forces one into the smart, but casual methodology, which then leads to the benefits listed above. ■